

0.- Escenario

En esta guía configuraremos dos equipos para que puedan conectarse entre ellos mediante ssh usando clave pública/privada en lugar de usuario y contraseña.

Partimos de dos máquinas virtuales limpias y de una configuración de red que permita llegar de un equipo a otro y viceversa.

1.- Motivación

Usar usuario y contraseña ya no se considera seguro. En un nivel superior tenemos el cifrado.

Los cifrados requieren de una clave. El tipo de cifrado adecuado y la clave correcta permiten encriptar y desencriptar datos sin problema. Un ejemplo muy simple es el cifrado César que consiste en desplazar cada letra un número determinado de posiciones en el alfabeto. El cifrado César con clave 1 de la palabra "HOLA" sería "IPMB". Podemos ver como conociendo la clave somos capaces de encriptar y desencriptar sin problema.

A este tipo de cifrado se le llama **simétrico** puesto que la misma clave sirve para encriptar en los dos sentidos. Son cifrados muy rápidos para lectura y escritura por lo que se suelen aplicar a discos.

Otro tipo de cifrado son los **asimétricos** que se caracterizan por tener una forma de encriptar y otra distinta no relacionables para desencriptar. En este tipo de cifrado usamos los conceptos de llave pública y llave privada.

Usando un ejemplo de la vida real podríamos decir que la cerradura de tu casa es tu llave o clave privada. Esa combinación concreta de mecanismos que permite abrir la puerta es algo solo tuyo, no lo puedes compartir. Aplicado a nuestro tutorial, la llave privada es a encargada de cifrar los paquetes con información, solo tú deberías ser capaz de cifrar información en tu nombre.

Volviendo al ejemplo del mundo real, también tenemos llaves que abren nuestra casa. Estas llaves podemos repartirlas entre todos aquellos a los que autorizamos. Es probable que en tu familia alguien tenga unas copias de tu casa por si las pierdes o simplemente porque quieres que entren cuando ellos quieran. Seguramente tú mismo fuiste el que repartió las copias. Esta será la llave pública.

Aplicado a nuestro tutorial, la llave pública debemos repartirla a todos aquellos ordenadores que queremos que desencripten los paquetes de información que enviamos.

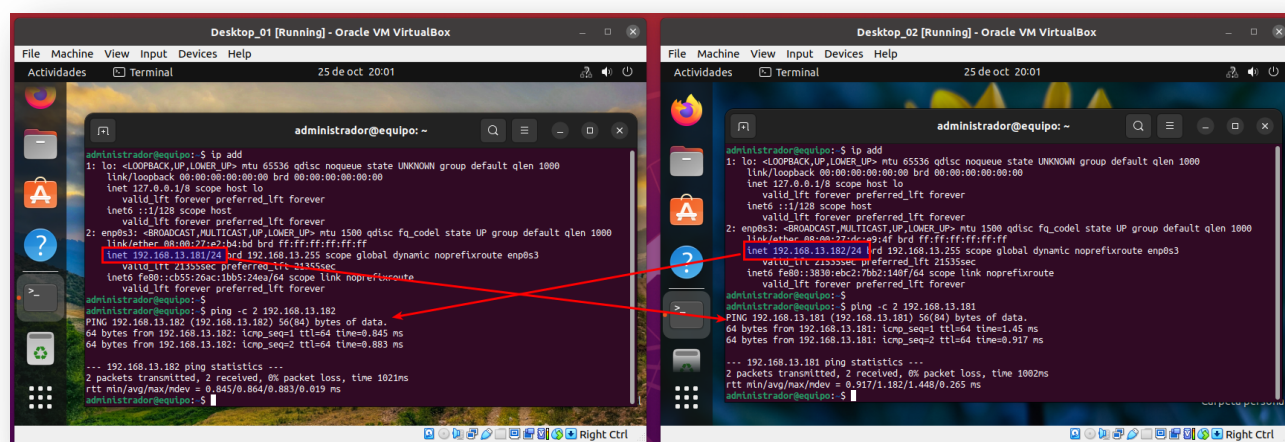
2.- Instalación servidor ssh

Empecemos con un ejemplo tradicional, una conexión ssh.

Las conexiones ssh nos permiten abrir sesiones remotas lo que nos permite conectarnos a múltiples equipos dispersos geográficamente y trabajar en ellos como si físicamente estuviéramos allí.

Probablemente en una red local doméstica no tenga mucho sentido, pero sí lo tienen en un entorno empresarial donde la seguridad es importante y los equipos están repartidos en distintas instalaciones.

En primer lugar, vamos a comprobar que ambos equipos navegan y se ven.



```
ip address
```

Con este comando conocemos qué dirección IP tenemos asignada en casa equipo

```
ping X.X.X.X
```

Con el comando ping probaremos si los equipos se ven. Si este comando falla y las direcciones IP están bien es probable que tengamos un problema con el firewall o con el enrutado en algún punto, pero este asunto se escapa al objetivo de esta guía.

En este caso estoy usando Ubuntu 22.04 Desktop que, aunque sí tiene instalado el cliente que nos deja conectarnos a otras máquinas, no tiene instalado el servidor SSH que permite conexiones desde remoto a nuestra máquina.

```
sudo apt install openssh-server
```

Con este comando instalamos el servidor SSH y desde ese momento, el puerto 22 de nuestra máquina quedará escuchando nuevas peticiones. Recordamos que en la ruta `/etc/ssh/sshd_config` tenemos el archivo de configuración del servidor ssh donde podremos cambiar el puerto de escucha, IPs permitidas y otros parámetros.

3.- Conexión ssh con contraseña

En un escenario tradicional, el equipo que actúa de cliente ejecutará el siguiente comando

```
ssh administrador@192.168.13.181
```

Este comando abre y establece una conexión ssh con el equipo servidor indicado por la ip usando el nombre de usuario indicado antes de la arroba. Recuerda que estamos estableciendo una sesión remota así que el nombre de usuario debe existir en la máquina servidora.

La primera vez que conectemos con un equipo remoto desconocido, nuestro cliente ssh detectará su huella y nos preguntará si estamos seguros. Aceptaremos y ese equipo se guardará en el archivo `/home/javi/.ssh/known_hosts`

En la siguiente captura podemos ver como estoy logueado con mi usuario *javi* en el equipo que se llama cliente y hago un ssh para conectarme al equipo llamado servidor con el usuario administrador.

```
javi@cliente:~$ ssh administrador@192.168.13.181
The authenticity of host '192.168.13.181 (192.168.13.181)' can't be established.
ED25519 key fingerprint is SHA256:0x4fsdtbj/696S3YYXxLJPC6zbUhIcKQYfSbSruFAM.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.13.181' (ED25519) to the list of known hosts.
administrador@192.168.13.181's password:
```

A efectos prácticos es como si tuviéramos dos terminales abiertos en el equipo servidor.

4.- Conexión ssh con claves privadas-públicas

Nuestros escenarios serán bastante más complejos y tendremos que gestionar distintos tipos de usuarios y nodos. Además, habrá que automatizarlos en la medida de lo posible y no queremos que se pare para preguntarnos una contraseña. La opción de guardar contraseñas en ficheros planos tampoco parece muy segura.

Para solucionar este problema usaremos ssh autenticando mediante pares de claves y así no será necesario usar usuario y contraseña. Es decir, la ventaja no es solo librarnos de la contraseña sino que también nos proporciona que todas las comunicaciones son seguras porque viajan cifradas y que garantizamos la autoría y la no modificación del paquete.

En primer lugar, es necesario generar las llaves pública y privada. Si recordamos el ejemplo de la cerradura (privada) y la llave (pública) y tenemos en cuenta que lo que se va a mandar es un paquete cifrado entonces parece claro que debemos generar las llaves en el equipo cliente.

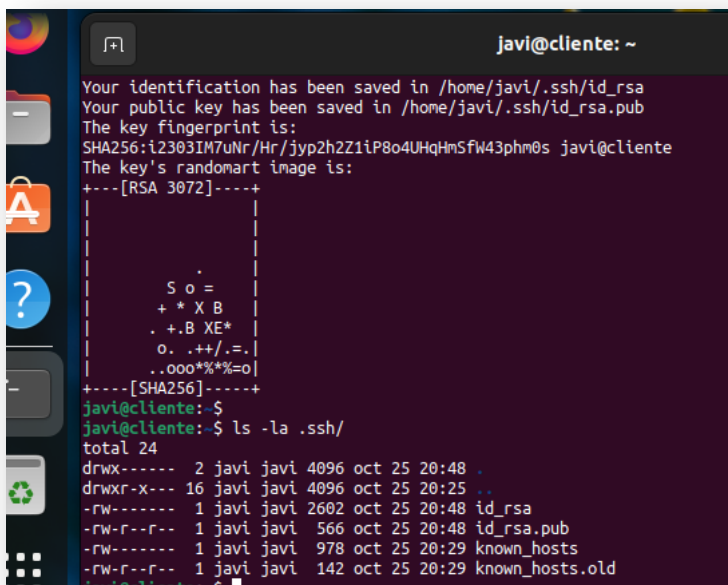
Pensemos esto despacio porque es importante. La primera acción que ocurrirá cuando todo esté funcionando es que el equipo cliente enviará un paquete cifrado al equipo servidor para intentar establecer con él una nueva conexión. Para ello el cliente cifra con su llave privada el paquete y lo envía. El servidor recibe este paquete cifrado y usa la llave pública para descifrarlo.

```
ssh-keygen
```

Con este comando generamos el par de llaves. Al ejecutar este comando sin parámetros nos pedirá algunos otros datos que podemos dejar en blanco como el nombre del archivo de las llaves o una contraseña.

```
ssh-keygen -t rsa -b 4096
```

También es posible usar el comando con modificadores como en este caso donde indicamos que queremos usar RSA de 4096 bits.



```
javi@cliente: ~
Your identification has been saved in /home/javi/.ssh/id_rsa
Your public key has been saved in /home/javi/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:i2303IM7uNr/Hr/jyp2h2Z1iP8o4UHqHmSfW43phm0s javi@cliente
The key's randomart image is:
+---[RSA 3072]-----+
|
|  S o =
| + * X B
| . +.B XE*
| O. .+/.=.
| ..000*%%=0
+---[SHA256]-----+
javi@cliente:~$
javi@cliente:~$ ls -la .ssh/
total 24
drwx----- 2 javi javi 4096 oct 25 20:48 .
drwxr-x--- 16 javi javi 4096 oct 25 20:25 ..
-rw----- 1 javi javi 2602 oct 25 20:48 id_rsa
-rw-r--r-- 1 javi javi 566 oct 25 20:48 id_rsa.pub
-rw----- 1 javi javi 978 oct 25 20:29 known_hosts
-rw-r--r-- 1 javi javi 142 oct 25 20:29 known_hosts.old
javi@cliente:~$
```

Dentro de la subcarpeta `.ssh` encontraremos los archivos `id_rsa` e `id_rsa.pub`. El primero es la llave privada que se usará para cifrar. Esta llave debe mantenerse a buen recaudo y no compartirla. El segundo archivo, el que acaba en la extensión `pub` es la llave pública que podemos repartir a todos los equipos que queremos que puedan descifrar nuestros paquetes.

En este caso queremos que el servidor pueda descifrar nuestros mensajes así que le copiamos la llave pública.

```
ssh-copy-id -i /home/javi/.ssh/id_rsa.pub administrador@192.168.13.181
```

Con este comando, la ruta de la llave pública del cliente y los datos para una conexión ssh al servidor copiamos la llave pública.

```
javi@cliente:~$ ssh-copy-id -i /home/javi/.ssh/id_rsa.pub administrador@192.168.13.181
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/javi/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
administrador@192.168.13.181's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'administrador@192.168.13.181'"
and check to make sure that only the key(s) you wanted were added.

javi@cliente:~$
```

En el equipo servidor se habrá creado o añadido nuestra llave pública al archivo
`/home/administrador/.ssh/authorized_keys`

```
administrador@servidor:~/.ssh$ cd .ssh/
administrador@servidor:~/.ssh$ ls -l
total 4
-rw-r----- 1 administrador administrador 566 oct 25 21:07 authorized_keys
administrador@servidor:~/.ssh$ ls -la
total 12
drwxr-xr-x 2 administrador administrador 4096 oct 25 21:07 .
drwxr-xr-x 16 administrador administrador 4096 oct 25 20:21 ..
-rw-r----- 1 administrador administrador 566 oct 25 21:07 authorized_keys
administrador@servidor:~/.ssh$ cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCA8FpPKKyL9/L+W5v6E3Xx022d3kBosGSpQo/k/C+RbLSf3mh1f6rHpz50iphMvm
8AiUa/I8xwUDfNAmteJ0M+nXGWMYcLaN1BXu93CULGU5Q1ecovuXfoWfCd+XC6uofx3Z6dSuoSdASmHkV7jxuI1rTK0Egf16e7Wfv
Mb5GcwzTFNSAGj+CBku2MfhKMTD9hNiV9Ji2Z9p9FhT/0cH8mYubYnLx2/XlgM8d4Aqsv8ilCZs94MRTdzWuHSVw8s9jLFM4t3KpiQ
MA/K41jV7YbgGD300XmdS2TRqIdX7mCq39dvqPiNxrT4xmgCGbVZHUyGUTai+L5gY1JTUwq++gTKCReLL8kxZagXoKF3Tij+5ZWrat
gYPnhIhIT1uFGx2910AHB50an7cueaLUwGLfnYYNkV+0eypSEtB6bZPmLXwP+nx4rI4wSjx2HgJdedJWLW8NNsJ43TKUDG2AXoCqZw
bDM+3MLHqJmOnWF3pbmpzSxZubYqKJyr4NPBg7Qrc= javi@cliente
administrador@servidor:~/.ssh$
```

Añadir una nueva llave pública a este archivo es simplemente anexar a continuación por ello, en algunos sitios también lo podéis ver con una redirección típica de Linux.

```
cat /home/administrador/Descargas/id_rsa.pub >> /home/administrador/.ssh/authorized_keys
```

A partir de este momento podemos conectarnos mediante ssh sin necesidad de usar usuario y contraseña algo que resulta ideal si queremos hacer scripts o automatizar tareas. Pensemos que en este curso nos preparamos para manejar clusters de nodos por lo que esta herramienta facilita y a la vez asegura más nuestras comunicaciones



**XUNTA
DE GALICIA**

CONSELLERÍA DE CULTURA,
EDUCACIÓN, FORMACIÓN
PROFESIONAL E UNIVERSIDADES



IES de Teis
Avda. de Galicia, 101
36216 – Vigo

Tfno: 886 12 04 64
e-mail: ies.teis@edu.xunta.es
<http://www.iesteis.es>



**FORMACIÓN
PROFESIONAL**



**XUNTA
DE GALICIA**



Financiado pola
Unión Europea
NextGenerationEU



Unión Europea
Fondo Social Europeo
O FSE inviste no teu futuro



GOBIERNO
DE ESPAÑA
MINISTERIO
DE EDUCACIÓN
Y FORMACIÓN PROFESIONAL



Plan de
Recuperación,
Transformación
y Resiliencia



Xacobeo 21-22